

WIDEBAND AND NARROWBAND SPECTRUM SENSING METHODS USING
SOFTWARE DEFINED RADIOS

by

Jason Stegman

B.S., Southern Illinois University, 2012

A Thesis

Submitted in Partial Fulfillment of the Requirements for the
Master of Science Degree

Department of Electrical and Computer Engineering
in the Graduate School
Southern Illinois University Carbondale
August, 2014

THESIS APPROVAL

WIDEBAND AND NARROWBAND SPECTRUM SENSING METHODS USING SOFTWARE DEFINED RADIOS

By

Jason Stegman

A Thesis Submitted in Partial
Fulfillment of the Requirements

for the Degree of

Master of Science

in the field of Electrical and Computer Engineering

Approved by:

Dr. Xiangwei Zhou, Chair

Dr. Haibo Wang

Dr. Ning Weng

Graduate School
Southern Illinois University Carbondale
June 6, 2014

AN ABSTRACT OF THE THESIS OF

Jason Stegman, for the Master of Science degree in Electrical and Computer Engineering, presented on June 6, 2014, at Southern Illinois University Carbondale.

**TITLE: WIDEBAND AND NARROWBAND SPECTRUM SENSING METHODS
USING SOFTWARE DEFINED RADIOS**

MAJOR PROFESSOR: Dr. X. Zhou

The ability to accurately sense the surrounding wireless spectrum, without having any prior information about the type of signals present, is an important aspect for dynamic spectrum access and cognitive radio. Energy detection is one viable method, however its performance is limited at low SNR and must adhere to Nyquist sampling theorem. Compressive sensing has emerged as a potential method to recover wideband signals using sub-Nyquist sampling rates, under the presumption that the signals are sparse in a certain domain. In this study, the performance and some of the practical limitations of energy detection and compressive sensing are compared via simulation, and also implementation using the Universal Software Radio Peripheral (USRP) software defined radio (SDR) platform. The usefulness and simplicity of the USRP and GNU Radio software toolkit for simulation and experimentation, as well as some other application areas of compressive sensing and SDR, is also discussed.

ACKNOWLEDGMENTS

I would like to thank Dr. Zhou for introducing me to this topic and giving me the tools necessary to conduct this research, and I also appreciate his valuable input and patience throughout the course of my graduate studies. I would also like to thank the other members of my thesis committee Dr. Haibo Wang and Dr. Ning Weng. Lastly, I thank Dr. Harackiewicz for allowing me to run some experiments in the Antennas lab, and thanks to Hemachandra Gorla for aiding me in setting up and running those experiments.

TABLE OF CONTENTS

Abstract	i
Acknowledgments	ii
List of Tables	v
List of Figures	vi
1 Introduction	1
2 Cognitive Radio	3
2.1 Spectrum Sharing Methods	3
2.2 Introduction to Cognitive	3
2.3 Spectrum Sensing	5
2.4 Resource Allocation	5
2.5 Cognitive Radio Applications	6
3 Spectrum Sensing	8
3.1 Spectrum Sensing in Multiple Dimensions	8
3.1.1 Temporal Holes	8
3.1.2 Opportunities in Frequency Domain	9
3.1.3 Opportunities in the Spatial Domain	9
3.2 Nyquist Rate Sensing Methods	10
3.2.1 Nyquist Sampling Theorem	11
3.2.2 Matched Filter	11
3.2.3 Energy Detection	12
3.3 Wideband Sensing Methods	12
3.3.1 Compressive Sensing	12
3.3.2 Filter Bank Spectrum Sensing	13
4 Software Defined Radio	14
4.1 Introduction to Software Defined Radio	14

4.2	USRP	14
4.2.1	USRP B210	15
4.3	GNU Radio	16
4.3.1	GNU Radio Functionality	16
5	Energy Detection Simulation and Implementation	18
5.1	Energy Detection Problem Formulation	18
5.2	Energy Detection Simulation	20
5.3	Energy Detection Implementation	25
5.3.1	Receiver	25
5.3.2	Transmitter	26
5.3.3	Experimental Results	26
6	Compressive Sensing Simulation and Implementation	29
6.1	Compressive Sensing Problem Formulation	29
6.2	Compressive Sensing Simulations	31
6.2.1	Compressive Sensing for Time Domain Recovery	31
6.2.2	Compressive Sensing for Frequency Domain Recovery	34
6.3	Compressive Sensing Implementation	42
7	Conclusion	47
	References	49
	Vita	53

LIST OF TABLES

6.1	Compressive sensing performance with $m = 1024$ and $n = 512$	38
6.2	Compressive sensing performance with $m = 1024$ and $n = 768$	42

LIST OF FIGURES

2.1	Spectrum occupancy in Chicago and New York City [1].	4
3.1	Channel hopping in the time and frequency domain [12].	10
3.2	Hidden primary user problem.	11
5.1	P_D vs. P_F at different SNR.	21
5.2	P_D vs. threshold at different SNR.	21
5.3	P_F vs. threshold at different SNR.	22
5.4	P_D vs. P_F at different values of N	23
5.5	P_D vs. threshold at different values of N	23
5.6	P_F vs. threshold at different values of N	24
5.7	P_D vs. P_F demonstrating the SNR wall limitation for energy detection.	25
5.8	Experimental setup.	26
5.9	P_D vs. P_F at different SNR.	27
5.10	P_D vs. γ at different SNR.	28
5.11	P_F vs. γ	28
6.1	Original Signal FFT.	32
6.2	Original Signal FFT.	33
6.3	FFT of recovered signal.	33
6.4	Recovered Signal.	34
6.5	Simulation signal generation block diagram.	36
6.6	Original spectrum with SNR = -1.94 dB.	37
6.7	Recovered spectrum with SNR = -1.94 dB and $n = 512$	39
6.8	Original spectrum with SNR = -10.23 dB.	39
6.9	Recovered spectrum with SNR = -10.23 dB and $n = 512$	40
6.10	Recovered spectrum with SNR = -10.23 dB and n is increased to 768.	41
6.11	Original spectrum with SNR = -13.06 dB.	41

6.12	Recovered spectrum with SNR = -13.06 dB and $n = 768$	42
6.13	Experimental setup.	43
6.14	Original spectrum.	45
6.15	Recovered spectrum.	45
6.16	Recovered spectrum.	46

CHAPTER 1

INTRODUCTION

The wireless radio spectrum is a resource that continues to see increased use as more devices go wireless. Under the current spectrum allocation scheme the Federal Communications Commission (FCC) in the United States licences out specific frequency bands for very specific uses. This causes many bands to face overcrowding and some bands to be heavily under-utilized. This introduces the need for more flexible spectral access methods.

Cognitive radio technology sets out to balance out the usage of the wireless spectrum. It does this by allowing unlicensed users to access licensed frequency bands, the only stipulation is that the unlicensed user must not cause harmful interference to the licensed user. This means that a cognitive radio must be able to sense what is going on around it at all time, this is called spectrum sensing. Once the radio determines what it is going on around it, it must then adapt its parameters to be able to access an open channel, this is called resource allocation. The focus of this thesis is on the spectrum sensing aspect of a cognitive radio.

In order to implement and also test these flexible access methods, there is a need for more flexible hardware that can support many different protocols and operate on a wide range of frequencies. Software defined radio (SDR) is a recent development that uses flexible hardware, that is defined by software, to define the radios operating parameters. This makes SDR useful for testing and implementing cognitive radio access methods.

The focus of this thesis is to simulate and test spectrum sensing methods for cognitive radio applications. Methods exist that can accurately detect a signal when knowledge of the signal of interest is known. However, a cognitive radio may not always know what type of signal is present, and thus needs to be able to detect unknown signals. Methods such as energy detection exist that do not require any knowledge of the signal of

interest. However, the detectable bandwidth is limited to that allowed by Nyquist sampling theorem, and thus faces practical constraints on the detectable bandwidth. For a cognitive radio to be effective, it should be able to sense a large bandwidth in a short amount of time. Compressive sensing is one method that aims to be able to sense wide bandwidths while sampling below Nyquist rate.

Both energy detection and compressive sensing are the two spectrum sensing methods studied in this paper. Both methods are simulated and also implementation using SDR. MATLAB and Python with GNU Radio toolbox are used for the simulations, and the Universal Software Radio Peripheral (USRP) SDR platform is used for implementation. Practical limitations of both methods are discussed and compared.

Chapter 2 discusses cognitive radio and some of its key characteristics. Chapter 3 then discusses spectrum sensing for cognitive radios as a multi-dimensional problem as well as some common spectrum sensing methods. Chapter 4 discusses SDR as the flexible technology that enables these methods to be implemented and tested. Chapter 5 discusses energy detection in greater detail, as well as the simulation and implementation results for energy detection. Chapter 6 discusses compressive sensing and its application to communications systems. Chapter 6 also discusses the simulation and implementation results for compressive sensing. Chapter 7 then concludes this report and discusses some ideas for future work.

CHAPTER 2

COGNITIVE RADIO

2.1 SPECTRUM SHARING METHODS

The wireless radio spectrum is a resource that has seen a significant increase in usage over the past few decades. Regulatory agencies such as the Federal Communications (FCC) in the United States are responsible for determining by whom and for what a given frequency band is used. Currently, frequency bands are divided up, and certain users or companies are given licenses by the FCC to transmit and receive in a given frequency band for a specific purpose. For instance, there are frequency bands that are used for cellular phones, satellites, marine radio, aviation radio, military communications, AM/FM Radio etc. Any usage in a frequency band by an unlicensed user is illegal, and the unlicensed user could face penalty of law.

This methodology has some inherent flaws. Some frequency bands are heavily utilized and face overcrowding, while some frequency bands are underutilized [1]. The utilization of frequency bands varies over different time slots and also varies across different geographic areas. For instance, marine radio bands are not utilized in geographic areas without large bodies of water, while cellular bands are very heavily utilized in populated areas. Figure 2.1 shows a plot of the average spectrum usage in Chicago and New York City. As seen in the plot, the frequency utilization is very uneven, some bands are very crowded while others are underutilized. An emerging technology called Cognitive Radio sets out to resolve this problem and make the frequency utilization more efficient.

2.2 INTRODUCTION TO COGNITIVE

The term Cognitive Radio was first coined by Joseph Mitola [2]. The FCC has defined it as follows, “A cognitive radio is a radio that can change its transmitter parameters based on interaction with the environment in which it operates.” [3] In other

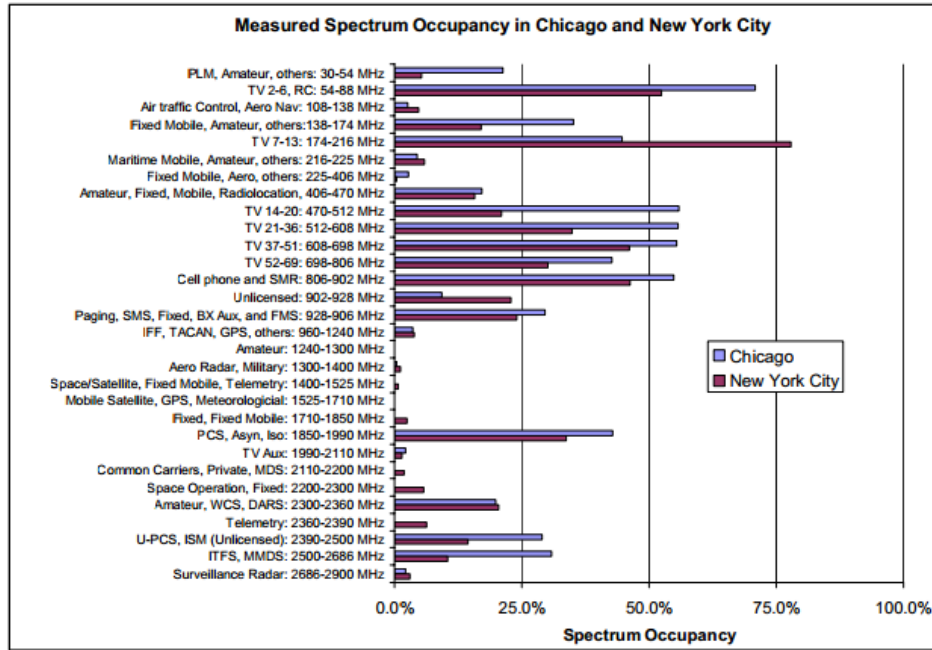


Figure 2.1. Spectrum occupancy in Chicago and New York City [1].

words a cognitive radio is a thinking radio that can sense its electromagnetic environment and then dynamically change its operating parameters to utilize the available wireless spectrum around it.

The purpose of this technology is to balance out the utilization of the wireless spectrum. This will in turn support more users with more wireless technologies. This is done by exploiting underutilized bands of the wireless spectrum. A user that is licensed or has higher priority to use a specific portion of the spectrum is called a primary user. An unlicensed user or a user with lower priority on a portion of the spectrum is called a secondary user. The secondary user utilizes cognitive radio technology to sense the presence or absence of primary users on a given frequency band, this is called spectrum sensing. If the primary user is present, the secondary user does not use the band to avoid causing interference to the primary user. If the primary user is not present, the secondary user adapts its operating parameters to utilize the given band, this is called resource

allocation. If the primary user returns to the band, the secondary user must vacate the band or adapt its operating parameters to avoid causing the primary user interference. Hence, the two main processes that a cognitive radio uses to exploit a frequency band are spectrum sensing and resource allocation.

2.3 SPECTRUM SENSING

Spectrum sensing is the process of determining whether or not a primary user is present in a given frequency band. When a frequency band is vacant, it is called a spectrum hole. Spectrum sensing sets out to determine where these holes are located. This process can be done with various methodologies that are discussed in greater detail in chapter 3.

Spectrum holes can occur in multiple dimensions. Some example dimensions where spectrum holes may exist are time, space, frequency, power, and code. For example a spatial hole comes about when a primary user is not present within a geographic area, or is far enough from a secondary user that the secondary users transmission will not cause the primary user interference. A temporal hole comes about when a primary user is not transmitting in a specific time slot. The secondary user then senses the absence of the primary user in time, and then exploits the vacant channel. In both scenarios it is imperative that secondary user can sense the return of the primary user so as to not cause them harmful interference.

2.4 RESOURCE ALLOCATION

Resource allocation in cognitive radio networks aims to minimize utilization of wireless resources while maintaining quality of service requirements [4]. Some common methods of doing so are power control, rate adaption, and optimizing various parameters using optimization methods. Past primary user behavior can also be taken into account in order to predict their future behaviour.

Resource allocation in cognitive radio networks differs from standard wireless networks in a few ways. In cognitive radio networks the available spectrum resources can fluctuate and vary within different areas of the cognitive radio network whereas in standard wireless networks the spectrum resources generally remain static. In addition to the secondary users, the primary users activity must also be taken into account. Power control is more important in cognitive radio networks because spatial reuse can allow for more users. [5]

Advanced resource allocation schemes use optimization techniques such as linear, non-linear, and dynamic programming [4, 6]. This is often done in a centralized manner which allows for greater computational complexity, however, this poses some network overhead cost and poses greater challenges to implementation. Some common objective functions are average data rate, throughput, and spectrum use efficiency. Some common constraints are hardware limitations, overall transmit power, minimum rate, and channel limitations [7].

2.5 COGNITIVE RADIO APPLICATIONS

Cognitive radio has some very important areas of application in today's world as well as looking into the future. Some of the immediate applications areas are TV white spaces, military use, and emergency response and public safety.

Effective communication is imperative for the next generation warfighter. Military forces must be able to communicate with multiple units that often require the use of many different radios with varying operating parameters to communicate. A software defined radio using cognitive radio technology can consolidate their communications needs into a single portable radio. Militaries have already invested in the development of this technology. [8]

Emergency responders must be able to communicate in extreme environments. Cognitive radio technology can enable first responders to effectively communicate with

each other and with survivors. The current communication infrastructure cannot currently meet the future demand [9].

Cognitive radio technology is already in use for exploiting TV white spaces. TV white spaces are unused frequencies between TV broadcasting channels that are used as a guard band. There are also unused frequencies that have recently become available due to the switch from analog to digital TV broadcasts. Cognitive radio technology is used to allow unlicensed users access to these bands for various purposes, such as the IEEE 802.22 protocol which sets out to create a standard for using TV white spaces for providing broadband access to rural areas [11].

CHAPTER 3

SPECTRUM SENSING

Spectrum Sensing is the process determining the presence or absence of a primary user. In order for a cognitive radio to be effective it must be able to operate without causing interference to a primary user. In this way spectrum sensing is the most important aspect of a cognitive radio because it determines the presence of the primary user, and it also must be able to detect the return of the primary user. This section will discuss the challenges of spectrum sensing as a multidimensional problem, and also the tradeoffs and methods of implementing spectrum sensing schemes for cognitive radio applications.

3.1 SPECTRUM SENSING IN MULTIPLE DIMENSIONS

Spectrum holes can appear in various dimensions. Each dimension has challenges to implementation and methods of overcoming these challenges. The primary three dimensions a spectrum hole may appear are time, space, and frequency. Other possible dimensions are the code and angle dimensions [12]. This paper will focus on some of the methods of finding spectral opportunities in the primary three dimensions.

3.1.1 Temporal Holes

As introduced in 2.3, a temporal hole is a spectrum hole in time, or determining when a primary user is absent in a certain slot of time. A temporal hole can be detected using either a single or dual radio configuration [12]. In a single radio configuration, the cognitive radio allocates a certain slot in time for sensing and the other portion of the time for transmitting. This limits the sensing accuracy, whereas a primary user may return and will not be immediately detected. It also decreases the spectrum utilization because a portion of the available time slot is spent sensing.

The other configuration is a dual radio configuration, where a cognitive radio can transmit and sense simultaneously. The drawback to this configuration is the increased hardware and power consumption, however, it increases the sensing accuracy and spectrum efficiency. Many methodologies [13], [14], [15] have set out to optimize the sensing time to create a balance between sensing time and detection accuracy.

3.1.2 Opportunities in Frequency Domain

A cognitive radio must be able to adaptably detect spectrum holes in the frequency domain. Since a primary user can return at any time and at any frequency, it should be able to efficiently detect a wideband of frequencies and accurately determine the presence and location of a primary user within the band, giving it the flexibility to quickly move from channel to channel when needed. This poses a challenge for hardware implementation.

The bandwidth a receiver can accurately sense is governed by Nyquist sampling theorem (see section 3.2.1). Detecting a wide band of frequencies requires a high rate ADC or a low rate ADC that must sequentially cycle through narrow bands one by one to detect a larger band. The high rate ADC can be expensive and require a large amount of power. The sequential method takes a longer amount of time which can reduce overall spectral efficiency and is also less accurate because a primary users activity may change while the channel is not being sensed. Other novel methods of sensing a wideband at a rate lower than the Nyquist rate are discussed in sections 3.3. Figure 3.1.2 shows how a cognitive radio can adaptively shift between spectral holes in both the time and frequency domain.

3.1.3 Opportunities in the Spatial Domain

Spectral opportunities can also become available in the spatial domain when a certain channel frequency is not utilized within a geographic area. This occurs from the

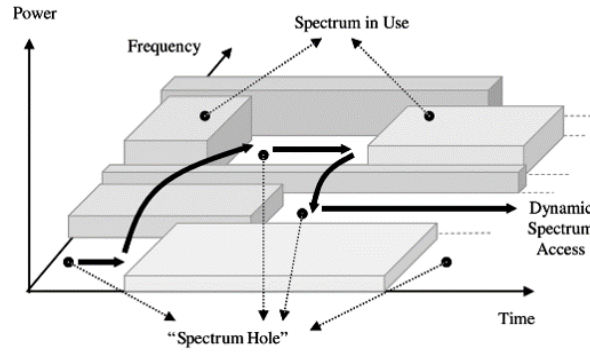


Figure 3.1. Channel hopping in the time and frequency domain [12].

signals propagation loss when transmitted over relatively large distances or geographic obstructions within a signal's path that degrade the signal. One of the challenges associated with this type spectral opportunity is called the hidden primary user problem. Figure 3.2 illustrates this. It occurs when a primary user transmitter is out of range of the secondary user receiver, and the primary user receiver is within the range of the secondary user transmitter or base station. The secondary users transmission causes the primary receiver costly interference.

One solution to this problem is to use cooperative sensing schemes [16], [17], [18], which allows secondary users to communicate spectral information to one another. One challenge to cooperative sensing is the network overhead cost and the cooperation amongst secondary users that is needed in order for them to communicate with one another.

3.2 NYQUIST RATE SENSING METHODS

The following section outlines some of the common methods used for spectrum sensing that adhere to Nyquist sampling theorem. One of the major differences between these methods are whether or not information about the signal needs to be known prior to detection.

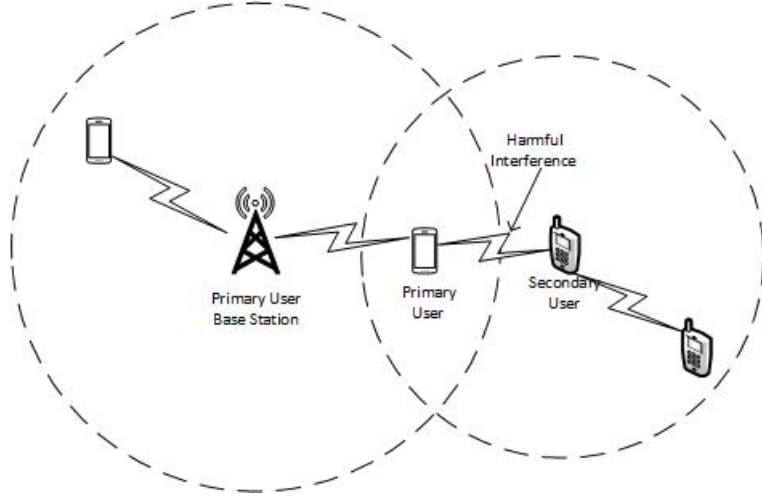


Figure 3.2. Hidden primary user problem.

3.2.1 Nyquist Sampling Theorem

According to Nyquist Sampling Theorem, in order for a signal to be fully recovered it must be sampled at or above the Nyquist frequency, f_s , which is

$$f_s > 2B, \quad (3.1)$$

where B is the signal bandwidth.

3.2.2 Matched Filter

When the secondary user knows what type of signal the primary user is transmitting, the optimum detection method is matched filtering [19]. Using matched filtering takes a short period of time for the signal to be detected and also produces a low probability of false alarm [19]. However, in order for the secondary user to use matched filtering, it must demodulate the signal, which requires knowledge of its frequency and modulation type. Matched filtering is often impractical because this information may not always be known. For a cognitive radio, it may also require a receiver with algorithms or hardware to detect many different modulation schemes which is costly in terms of hardware, power consumption, and complexity.

3.2.3 Energy Detection

Energy detection is the most widely used method of spectrum sensing due to its computational simplicity and ease of implementation compared to other methods [9]. Energy detection operates by sampling a signal and then computing the energy of the received samples, it then compares the signal energy to a pre-determined threshold to determine the presence or absence of a signal. Hence, no knowledge of the signal of interest is required for energy detection.

The uncertainty of noise power limits the performance of energy detection [19]. In low SNR environments, when the primary user signal falls near or below the noise floor, the performance of energy detection suffers even when the sensing duration is increased. This is called the SNR wall limitation. Many variations on the energy detector exist, such as differential energy detection [20], and an improved energy detection algorithm in [21]. However these methods increase computational complexity and in [21] require a priori knowledge of the noise power. Chapter 5 discusses the energy detection algorithm in greater detail and presents the simulation and implementation results.

3.3 WIDEBAND SENSING METHODS

3.3.1 Compressive Sensing

Compressive sensing aims to allow signal recovery by solving a system of under-determined equations. It does this by exploiting the sparsity of a signal in a particular domain, and solving a convex optimization problem. Compressive sensing has many applications, from astronomy to image processing. The main application for communications systems, is for recovering wide-band signals at sub-Nyquist rates. Normally, in order for a signal to be fully recovered it has to be sampled at or above the Nyquist frequency.

Sampling wideband signals becomes challenging when using conventional Analog to Digital Converters (ADC). Based on Nyquist's theory a signal must be sampled at twice

the bandwidth of a signal. In wideband applications the bandwidth is large and hence the sampling rate must be twice this. This is often not practical or too expensive using conventional ADC due to the very high sampling frequency [22]. For example if a signal with a bandwidth of 1 GHz is to be sampled, an ADC would need to sample at 2 GHz.

One challenge to the implementation of compressive sensing is the fact that the measurements need to be taken at random. It is somewhat impractical and difficult to implement an ADC that does not sample at a constant rate. Regardless of how many samples are taken, the minimum time between samples in an ADC will be limited.

3.3.2 Filter Bank Spectrum Sensing

Other wideband sensing methods [23], [24], aim to split up the signal into many smaller sub-bands using multiple bandpass filters, called filter banks. This requires a large amount of hardware which can be impractical to implement and expensive. These methods may also require multiple ADC's to sample each band which can cause a large amount of power consumption.

CHAPTER 4

SOFTWARE DEFINED RADIO

4.1 INTRODUCTION TO SOFTWARE DEFINED RADIO

A software defined radio (SDR) is a radio that performs a large portion of the signal processing using software rather than hardware. Typically, a radio utilizes specialized hardware (mixers, amplifiers, modulators, demodulators) in order to process a signal [25]. There are many applications and advantages of using SDR. One of the biggest advantages is its reconfigurability. Where some radios may become out of date due to the use of newer communication protocols or out of date hardware, the reconfigurability of a SDR can allow a radio to simply update its software in order to keep up to date with a newer protocol in use. This can be especially useful when a communication system is part of a larger system such as an airplane or vehicle. Rather than having to replace expensive hardware to meet new communications demands, a software update can be a quick and easy fix.

Another application of SDR is for academic and research use. An efficient SDR can support many different communication protocols, which makes it a great learning tool. SDR is also useful for research, because researchers can quickly implement and test existing or experimental communication protocols using SDR. All of these characteristics make SDR's an ideal candidate for the testing and implementation of cognitive radio technology.

4.2 USRP

The Universal Software Radio Peripheral (USRP) is a software defined radio platform produced by Ettus Research. The USRP is widely used in academic and commercial environments due to its relatively low cost and ease of use. Most of the USRP's have the ability to interface with a computer and utilize open source or widely

used commercial software packages such as GNU Radio, Lab View, or Simulink. This can turn a USRP and a computer into a high bandwidth radio.

The various USRP models follow a similar architecture, they all contain a flexible RF frontend for analog operations such as filtering and up/down conversion. The RF frontend connects to a motherboard that contains an ADC/DAC, FPGA, and an interface to connect with the host computer processor. Most of the high frequency digital signal processing such as digital filtering, interpolation, and decimation is done using the FPGA. Depending on the model, the USRP connects to the host computer in different ways. Some models use USB 2.0 or 3.0, there are network connected models that use ethernet, and some models contain an embedded processor with a built in Linux environment.

All of the models aim to do all of the waveform specific signal processing such as modulation and demodulation using the host processor. All of the analog RF and general purpose IF band processing such as analog/digital conversion or filtering, up/down conversion, decimation, and interpolation is done using the RF frontend and the USRP motherboard.

All of the USRP's use a driver software package called USRP Hardware Driver (UHD) that is compatible on Linux, Windows, and Mac. UHD contains functions for controlling all of the major parameters on the USRP such as gain, frequency, sample rate etc. UHD can easily work with software packages such as GNU Radio (section 4.3.1), LabView, or simulink, which in combination with a USRP and UHD, create a high bandwidth software radio [26].

4.2.1 USRP B210

The USRP used for the experimentation in this paper is the USRP B210. The B210 is a fully integrated single board. The RF front end consists of an AD9361 RFIC chip made by Analog Devices. This provides coverage from 70 MHz-6 GHz with up to 56 MHz of real time bandwidth. The RFIC chip also contains a flexible rate 61.44 MS/s ADC and

DAC with 12 bit resolution. The board also features a Xilinx Spartan 6 FPGA that is fully reconfigurable. The B210 is fully bus powered and interfaces with a host computer using high speed USB 3.0 that can provide up to 61.44 MS/s data transfer rate in one direction. The B210 has 2 transmit and 2 receive channels which are capable of operating in full-duplex mode simultaneously.

The B210 is the first USRP model to offer USB 3.0 compatibility and is also the first to be fully integrated onto a single bus powered board. This gives it portability and data rates comparable to Ethernet making it ideal for experimentation. The B210 is currently only supported by the latest version of UHD, which only has support on Linux and Mac. The experimentation in this report was all performed using Ubuntu and Debian systems with the GNU Radio software package.

4.3 GNU RADIO

For the experimentation in this paper, the software package used by the host computer is GNU Radio. GNU Radio is a free and open source software package that provides signal processing blocks used in the implementation of software radios. The software can run stand alone for simulation, or interfaced with hardware to create a software defined radio system.

4.3.1 GNU Radio Functionality

When interfaced with the USRP, GNU Radio collects or sends streams of digital data to and from the USRP. The digital data is processed using GNU Radio with built in signal processing “blocks”. These “blocks” are written in C++ and perform the critical signal processing functions such as modulation, encoding, filtering etc. GNU Radio programs are typically written in Python. The user can then tie together the necessary C++ blocks as well as any other data processing in a Python script. This allows users to quickly create and configure a program [26].

GNU Radio also has the functionality to plot data in real time. These functions can easily be tied into a user's Python program. GNU Radio also contains a graphical user interface called GNU Radio Companion (GRC) which allows user to graphically connect together signal processing blocks, plotting functions, and file operations. The GRC then automatically compiles a Python script that connects and executes the program.

CHAPTER 5

ENERGY DETECTION SIMULATION AND IMPLEMENTATION

In this section energy detection is simulated, then implemented and tested. The simulations are done using Monte Carlo trials in MATLAB. The algorithm is then implemented using Python and Gnu Radio with the USRP. The energy detector's performance is evaluated under varying conditions. The effect of increasing or decreasing the sensing time is analyzed, as well as the performance in different SNR environments.

5.1 ENERGY DETECTION PROBLEM FORMULATION

Energy detection is the most widely used method of spectrum sensing due to its computational simplicity and ease of implementation compared to other methods [9]. Energy detection operates by sampling a signal and then computing the energy of the received samples, it then compares the signal energy to a pre-determined threshold to determine the presence or absence of a signal. Hence, it does not require any prior information about the primary user signal.

For a received signal $y(n)$, energy detection decides between two hypotheses

$$y(n) = \begin{cases} s(n) + w(n) & : H_1, \\ w(n) & : H_0, \end{cases} \quad (5.1)$$

where $s(n)$ is the primary user signal, and $w(n)$ is additive white Gaussian noise (AWGN). Thus, hypothesis H_1 implies the presence of the primary user signal, and H_0 implies the absence of the primary users signal.

The decision statistic, $T(y)$, used for energy detection is the energy of the received samples, which is

$$T(y) = \frac{1}{N} \sum_{n=1}^N |y(n)|^2, \quad n = 1, \dots, N - 1, \quad (5.2)$$

where N is the number of signal samples taken. The decision statistic is then compared to a threshold, γ . The hypothesis test for energy detection is then

$$\text{if } \begin{cases} T(y) > \gamma & \text{decide } : H_1 . \\ T(y) < \gamma & \text{decide } : H_0 . \end{cases} \quad (5.3)$$

The most important performance metrics for the detection algorithms are the probability of detection P_D , and the probability of false alarm P_F . The probability of detection is the probability of accurately detecting the presence of a signal, and can be formulated as

$$P_D = Pr(T(y) > \gamma | H_1) . \quad (5.4)$$

The probability of false alarm, is the probability of incorrectly deciding H_1 (primary user present) when the primary user is actually absent. It can be formulated as

$$P_F = Pr(T(y) > \gamma | H_0) . \quad (5.5)$$

P_D should be maximized to ensure the best chance of accurately detecting the signal, while P_F should be kept as small as possible to minimize missed spectrum use opportunities. P_D and P_F can be optimized by adjusting the value of the decision statistic.

A bandpass filter is typically placed before the energy detector to remove any out of band components and prevent them from contributing to the decision statistic. When there is no bandpass filter, the FFT with a windowing function is typically used to help filter out any out of band components. The decision statistic for energy detection then becomes

$$T(y) = \frac{1}{N^2} \sum_{k=1}^N |X_f(k)|^2 , \quad (5.6)$$

where $|X_f(k)|$ is the magnitude of the N point FFT bins of $y(n)$.

The uncertainty of noise power limits the performance of energy detection [19]. In low SNR environments, when the primary user signal falls near or below the noise floor,

the performance of energy detection suffers even when the sensing duration is increased [9]. This is called the *SNR* wall limitation. This may lead to a high P_F or a low P_D .

5.2 ENERGY DETECTION SIMULATION

In order to simulate the energy detector algorithm, Monte Carlo trials were used. The threshold value, γ , is swept through a range of values. M simulations are then done at each threshold value. The presence and absence of the primary user signal is fluctuated by generating a random stream of zeroes and ones, and then multiplying this stream by the primary user signal, thus acting as a switch turning on or off the primary user signal. The probability of detection and probability of false alarm are then calculated based on the results.

The primary user signal is represented by a Gaussian random signal with zero mean, variance of σ_s . The noise is represented by a Gaussian random noise with zero mean and variance of σ_n . The SNR is then swept through a discrete range of values, and the P_D vs. γ , P_F vs. γ , and P_D vs. P_F are then plotted for each SNR value. The SNR for each case is given by

$$SNR = 10 * \log_{10} \frac{\sigma_s}{\sigma_n} . \quad (5.7)$$

The SNR value is modified in these simulations by changing the variance of the noise. The number of samples, N , used to compute each decision statistic is held constant at $N = 128$. The number of Monte Carlo trials done at each point is $M = 80000$.

Figure 5.1 shows the P_D vs. P_F plot for each SNR value used. It is easily seen from the plot that as the SNR increases, the detection probability increases and the probability of false alarm decreases. Figure 5.2 and figure 5.3 shows the plot P_D vs. γ and P_F vs. γ respectively. As the SNR decreases, the threshold required to achieve a certain P_D or P_F is increased.

The SNR is then held constant at -3 dB, and the number of samples, N , used in each computation of the decision statistic is then swept through six discrete values to show the

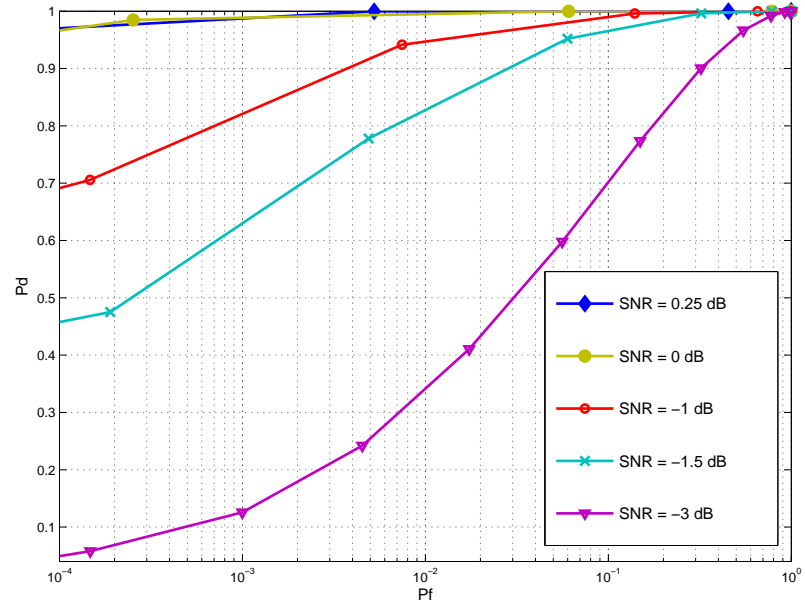


Figure 5.1. P_D vs. P_F at different SNR.

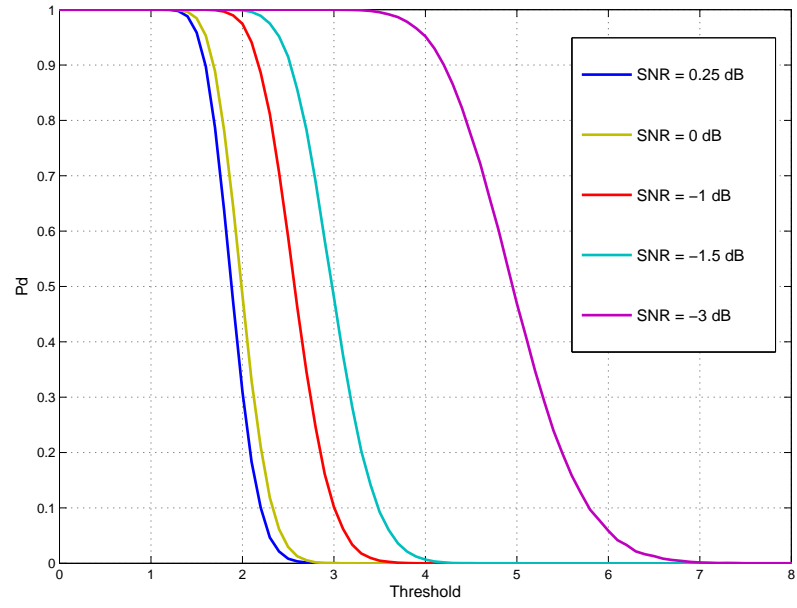


Figure 5.2. P_D vs. threshold at different SNR.

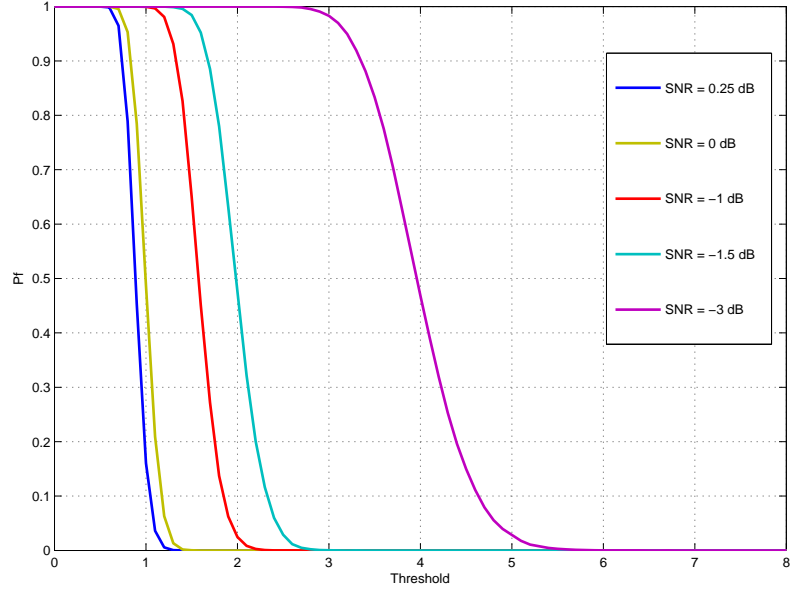


Figure 5.3. P_F vs. threshold at different SNR.

effect of increasing the sampling time. P_D vs. γ , P_F vs. γ , and P_D vs. P_F are then plotted for each value of N . In practice the sampling time, τ , is given by

$$\tau = \frac{N}{f_s}, \quad (5.8)$$

where f_s is the sampling frequency. Thus an increase in N correlates to an increase in the sampling time.

Figure 5.4 shows the P_D vs. P_F for different values of N , with the SNR fixed at -3 dB. As shown in the plot, a higher sampling time yields an increase in the P_D and a decrease in P_F . Figure 5.5 and figure 5.6 shows the P_D vs. δ and P_F vs. δ plots respectively, for varying values of N . An increase in N corresponds to an increase of the slope of the corresponding plot. At low SNR, increasing the sampling time will increase the sensing accuracy. The least accurate result from figure 5.1 is when the SNR is equal to -3 dB and N is fixed at 128 samples. Figure 5.4 shows that when N is increased to 1024 from 128, it results in a drastic increase in sensing accuracy.

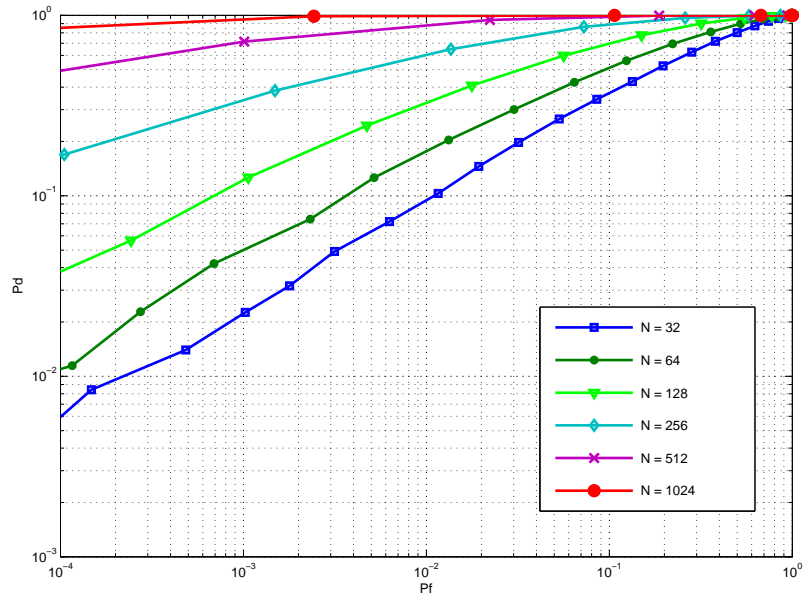


Figure 5.4. P_D vs. P_F at different values of N .

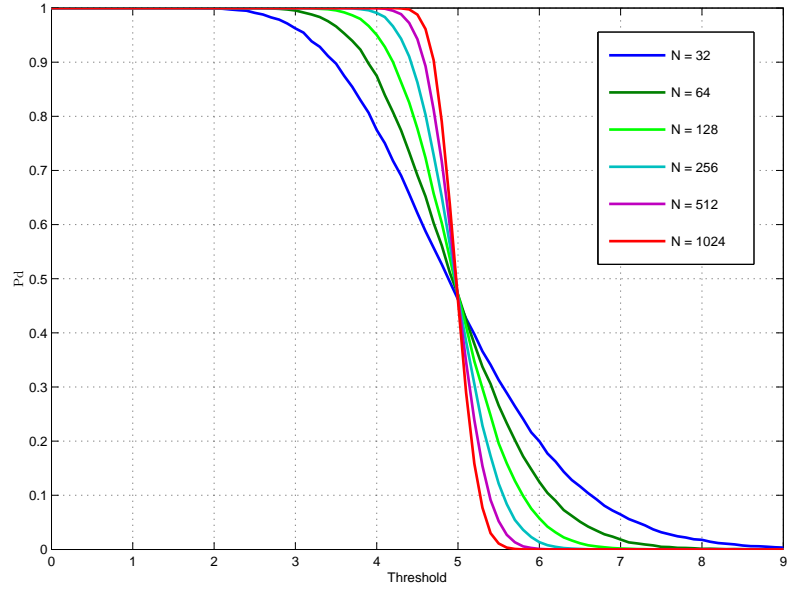


Figure 5.5. P_D vs. threshold at different values of N .

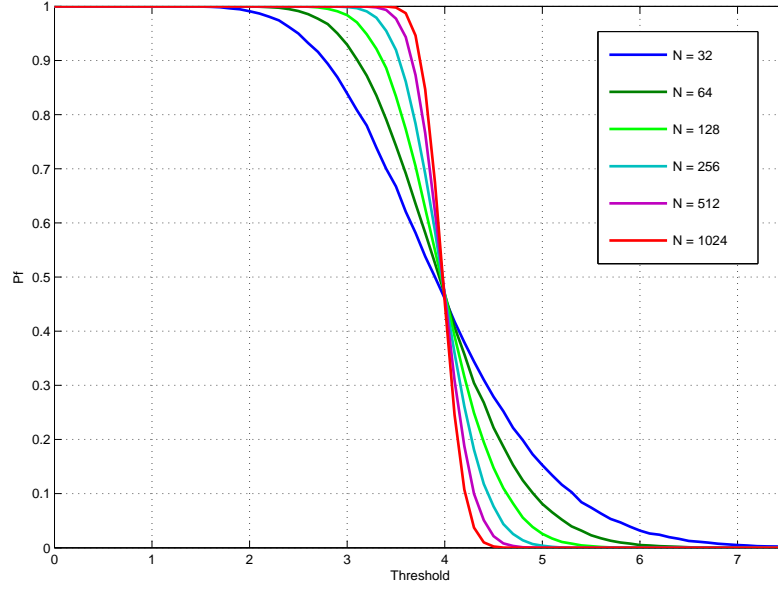


Figure 5.6. P_F vs. threshold at different values of N .

When the SNR is very low, the increase in sensing time will yield only a small increase in sensing accuracy. This is called the SNR wall problem for the energy detector. Figure 5.7 shows the P_D vs. P_F plot when the SNR is -6.43 dB, and N is stepped from 128 samples up to 4096 samples. When 4096 samples are used, P_D does not go above 0.9 and the P_F does not fall below 0.1. A P_D below 0.9 and P_F above 0.1 is likely unacceptable for most purposes.

In practice, increasing the sensing time introduces a tradeoff. In a single radio configuration, if a radio is sensing, it is wasting an opportunity to be utilizing a spectrum hole. In a dual radio configuration, increasing sensing time will increase the amount of power used by cognitive radio. When a wide band needs to be sensed, an energy detector may need to sequentially cycle through narrow bands in order to cover a large bandwidth. This causes an increase in overall sensing time.

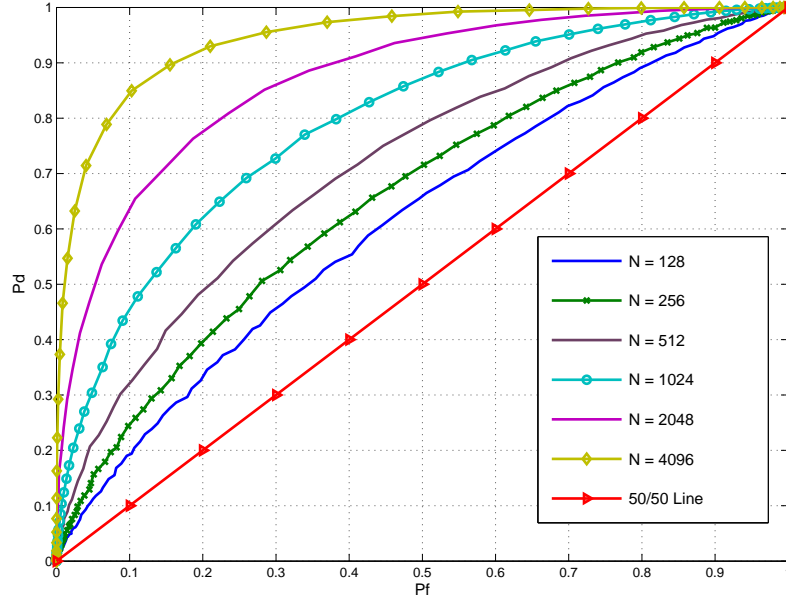


Figure 5.7. P_D vs. P_F demonstrating the SNR wall limitation for energy detection.

5.3 ENERGY DETECTION IMPLEMENTATION

In order to implement the energy detector, GNU Radio and the USRP software defined radio platform were used. Energy detection implementation is studied in [27], [28]. The results in this report focus on the performance at different SNR.

5.3.1 Receiver

The receiver was constructed using a USRP B210 connected to a laptop. The energy detection is done using a GNU Radio program written in Python. The energy detector is implemented using equation 6. The energy detection program written for these experiments is a modification GNU Radio's `usrp_spectrumsense.py` program that outputs the magnitude squared of the FFT points. The energy detection program then computes the decision statistic and compares it to a threshold. The program cycles through a specified range of threshold values, and computes M decision statistics at each threshold

value. The program outputs a file with the corresponding probability of detection or false alarm at each threshold value.

5.3.2 Transmitter

A USRP B100 and a laptop computer were used to construct the transmitter in these experiments. The laptop utilizes the GNU Radio Companion (GRC), which is a graphical user interface provided with GNU Radio that allows users to quickly build a software defined radio system. GRC is used to construct a system that transmits a BPSK modulated waveform centered at 1.5 GHz and/or white Gaussian Noise. The USRP's are connected together using an SMA cable. Since very little noise is present in this configuration, an artificial noise source is added at the transmitter side to produce a lower SNR environment. Figure 5.8 shows experimental setup used for the experimentation.

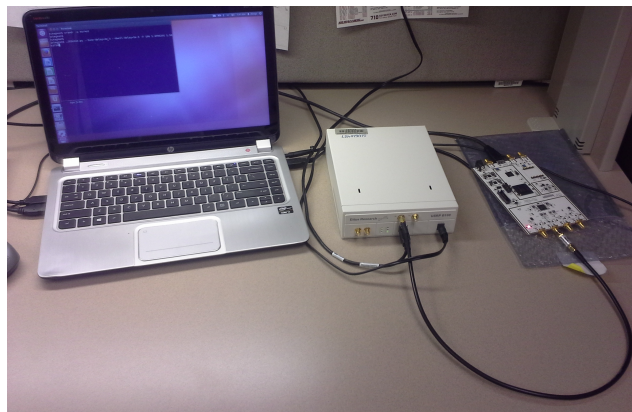


Figure 5.8. Experimental setup.

5.3.3 Experimental Results

The implemented energy detector is tested using a method similar to that of the simulations. Rather than cycling the primary user signal on and off, the noise only and signal plus noise cases are tested individually.

The transmitter first transmits the simulated noise only, so the receiver receives the

simulated noise plus the small amount of channel noise. The program cycles through threshold values and computes $M = 1000$ decision statistics at each threshold value. The P_F for that threshold value is then computed by dividing the number of times the received energy was greater than the threshold divided by M .

P_D is then computed by transmitting the signal plus the noise. The program again computes $M = 1000$ decision statistics at each threshold value. P_D is then calculated by dividing the number of times the received energy was greater than the threshold divided by the M .

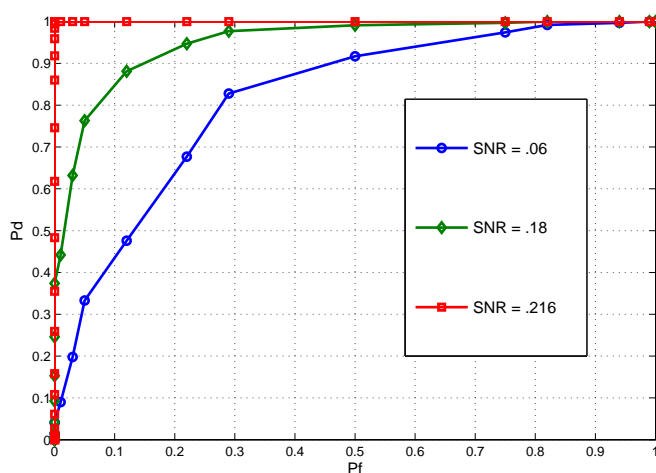


Figure 5.9. P_D vs. P_F at different SNR.

P_D is computed at three separate signal strengths. Figure 5.9 shows the plot of P_D vs. P_F for each of the three SNR's. As expected the detection becomes more accurate as the SNR increases. Figure 5.10 and figure 5.11 show P_D vs. γ and P_F vs. γ respectively.

The plot of P_F vs. γ shown in figure 5.11 remains relatively stable regardless of the SNR. This is because the noise level is not changed for the implementation experiments.

As the signal SNR is increased, the P_D vs. γ curves shift to the right, meaning a higher threshold is needed to obtain a certain P_D . This is the opposite of figure 5.2 which was obtained during the simulations. This is because in the simulations, the noise was

increased rather than the signal strength. An increase in noise corresponds to a decrease in SNR, however, it results in overall larger received energy and decision statistic. In the case of the implementations, an increase in signal strength increases the SNR and the total received energy or decision statistic, which results in the need for a higher threshold value.

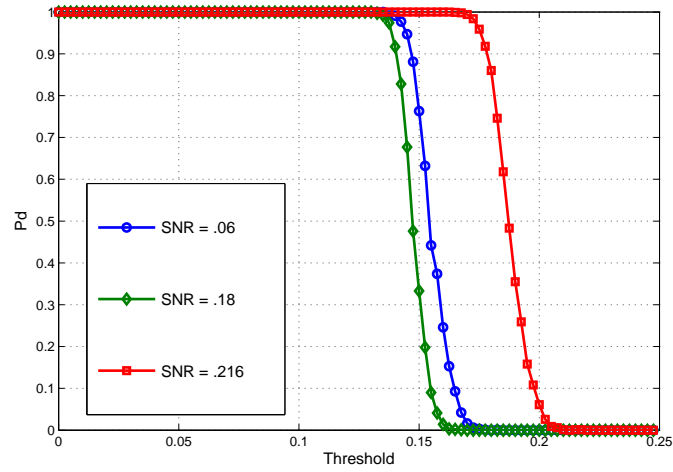


Figure 5.10. P_D vs. γ at different SNR.

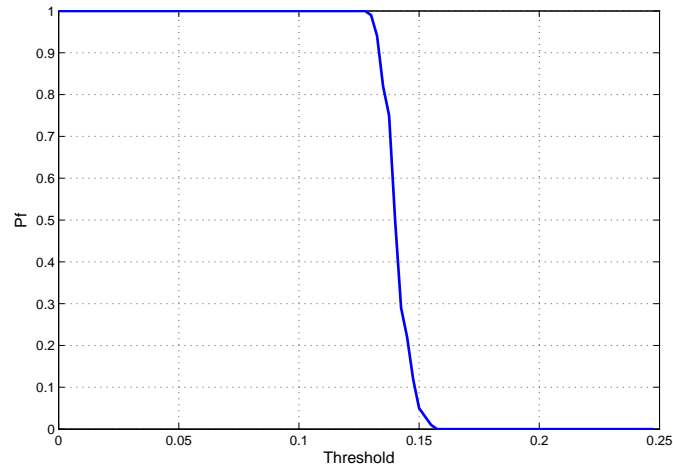


Figure 5.11. P_F vs. γ .

CHAPTER 6

COMPRESSIVE SENSING SIMULATION AND IMPLEMENTATION

In this section, compressive sensing theory is discussed in detail. The performance and some of the limitations to compressive sensing for communications systems are then analyzed through simulation and implementation. The simulation is done using MATLAB and Python. The implementation is done using USRP's.

6.1 COMPRESSIVE SENSING PROBLEM FORMULATION

Compressive sensing sets out to exploit the sparseness of a signal in a certain domain. A signal is said to be k -sparse if it has at most k nonzero elements. For instance a discrete signal of interest, z , may have sparse representation in the basis Ψ . Thus making

$$x = \Psi z , \tag{6.1}$$

where x is the sparse representation of z in the domain of Ψ . A wideband signal is often sparse in the frequency domain or in other words it has relatively few non-zero frequency components [29].

Based on the theory given in [30], compressive sensing allows an undetermined system of equations,

$$y = Ax , \tag{6.2}$$

to be solved. Where y is a set of measurements of the signal of interest, z , and x is a k -sparse unknown matrix representing z in the domain of Ψ , and A is an encoding or sampling matrix. The matrix A is an $m \times n$ matrix, matrix y has n elements, and the matrix x has m elements where $n \ll m$. Hence, there are far fewer samples in y than there are values in the undetermined matrix x . Solving x initially seems impossible, because there are many more unknowns than known variables. However it has been proven in [30] the underdetermined system can be solved that if the matrix A meets

certain requirements, and the matrix x is sparse, the system can be solved.

The matrix x can be recovered by finding the sparsest matrix x that solves the system, meaning that the solution with the fewest nonzero elements in a particular domain solves the underdetermined system of equations. The sparsest solution is found by minimizing the l_0 -norm as follows,

$$\min \|x\|_{l_0} \text{ s.t. } Ax = y . \quad (6.3)$$

Hence it finds x such that it contains the least amount of non-zero elements. However, it has been found that finding the l_0 -norm is not computationally feasible, so the l_1 -norm can be found instead, which is a convex optimization problem that can easily be solved using methods readily available [29]. The problem is then

$$\min \|x\|_{l_1} \text{ s.t. } Ax = y . \quad (6.4)$$

If the desired signal is sparse in a certain domain, typically frequency for communications systems applications, the samples must be taken in a different domain, typically time. This is an important mixing property. As stated by Terrence Tao, the measurement matrix must mix up all the basis vectors of data together.

In order for the above computation to successfully recover x , the matrix A must meet the restricted isometry property (RIP) which was introduced in [31]. The RIP constraint states that there must exist a constant δ_s , such that

$$(1 - \delta_s)\|x\|_{l_2}^2 \leq \|Ax\|_{l_2}^2 \leq (1 + \delta_s)\|x\|_{l_2}^2 . \quad (6.5)$$

The RIP says that the matrix A preserves the distance between any two k -sparse vectors, when they are mapped to the lower dimension of A . As determined in [32], determining if a matrix A satisfies the RIP is difficult, and thus it has been shown that there are three types of matrices that have a high probability of satisfying the RIP. The first type is a sub-gaussian matrix where all of the elements are i.i.d. Gaussian distributed, the second

case is a Bernoulli matrix where all of the elements are i.i.d. binary distributed, and the third case is when the matrix A is a random Fourier sub-matrix. Hence in the third case, A can be random rows of a Fourier matrix. In communications applications of compressed sensing, this is the most useful case.

In the presence of noise the problem can be adapted to solving $y = Ax + e$, where e is the error term. The problem is then formulated as:

$$\min \|x\|_{l_1} \text{ s.t. } \|Ax - y\|_{l_2} \leq \epsilon, \quad (6.6)$$

where ϵ is the magnitude of the error vector. In the presence of noise, the signal will contain many small but non-zero entries, however it has been shown in [30] that the sparsest solution is still recoverable in the presence of noise.

6.2 COMPRESSIVE SENSING SIMULATIONS

Compressive sensing theory is then applied to sparse signals in the frequency domain. Section 6.2.1 uses MATLAB to analyze examine the ability of compressive sensing to recovering signals in the time domain. Section 6.2.2 focuses on signal recovery in the frequency domain using Python and GNU Radio.

6.2.1 Compressive Sensing for Time Domain Recovery

Compressive sensing is applied to a simulated signal in a noiseless environment to examine its operation. In order to apply compressive sensing to the noiseless signal, the optimization problem of equation 3.11 must be solved, where the vector x with the smallest l_1 -norm is found. This is a basis pursuit problem, which is solved in these scenarios using l_1 -Magic from [33].

The simulated signal to be recovered is the summation of three sinusoids with frequencies of 1 kHz, 1.9 kHz, and 3.95 kHz at amplitudes of 1, 0.4, and 0.7 respectively. The signal is sampled at 8.192 kHz, and then plotted in both the time domain and the

frequency domain, shown in figure 6.1 and figure 6.2 respectively. From the figure, the signal is clearly sparse in the frequency domain, consisting of just three sinusoidal spikes, and the rest of the components are zero over a bandwidth of 4.096 kHz. In the time domain the signal is very densely populated with non-zero elements. Hence, in order to recover this signal with compressive sensing, the signal must be randomly sampled in the time domain, in order to recover the sparsest possible frequency domain representation of the samples, which corresponds to the vector x . The length of the original signal is

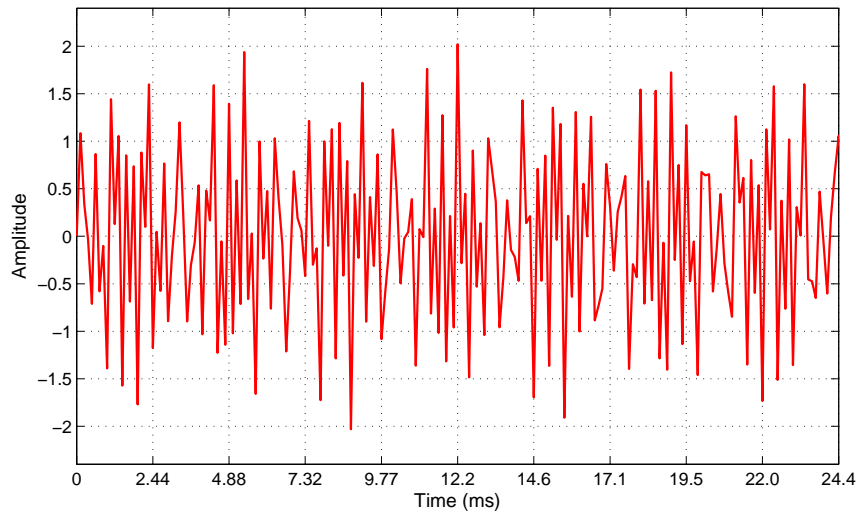


Figure 6.1. Original Signal FFT.

$m = 8.192k$ samples. Since the signal is sparse in the frequency domain, random rows of the inverse DFT matrix are used as the measurement matrix, A . The selected rows correspond to the random sample in the time domain, i.e. if sample 2 of the m samples is chosen in the time domain, row 2 of the $m \times m$ inverse DFT matrix will be used in the order that they are selected. In order to fully recover the sparse vector x , $n = 1024$ random samples of the time domain signal are taken. This yields an overall compression ratio of 0.2.

Compressive sensing is applied to the original time domain signal. The recovered signal in the frequency domain is shown in figure 6.3. The inverse FFT can be applied to

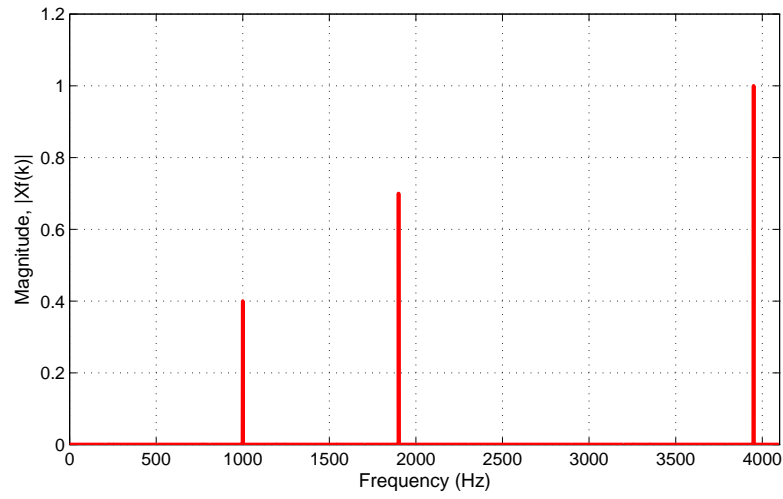


Figure 6.2. Original Signal FFT.

the frequency domain signal to recover the time domain signal which is shown in figure 6.4. As seen from the figures, compressive sensing is able to accurately recover the signal in both time and frequency domain using only 1024 samples.

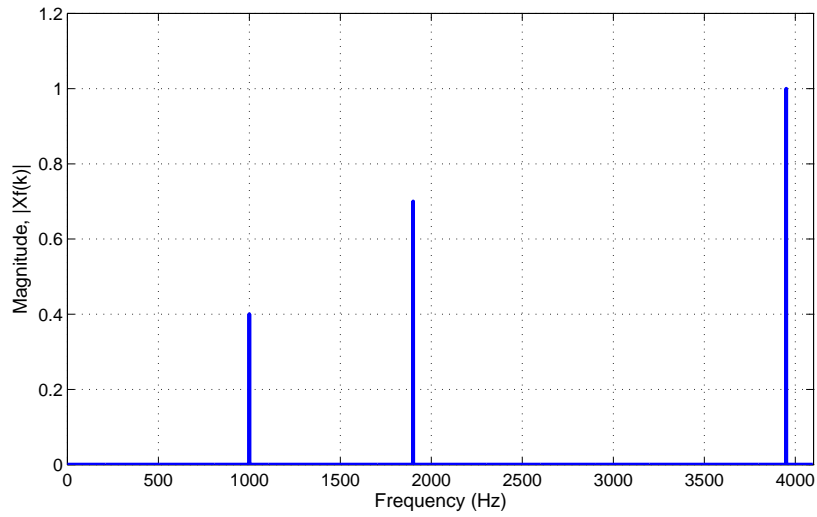


Figure 6.3. FFT of recovered signal.

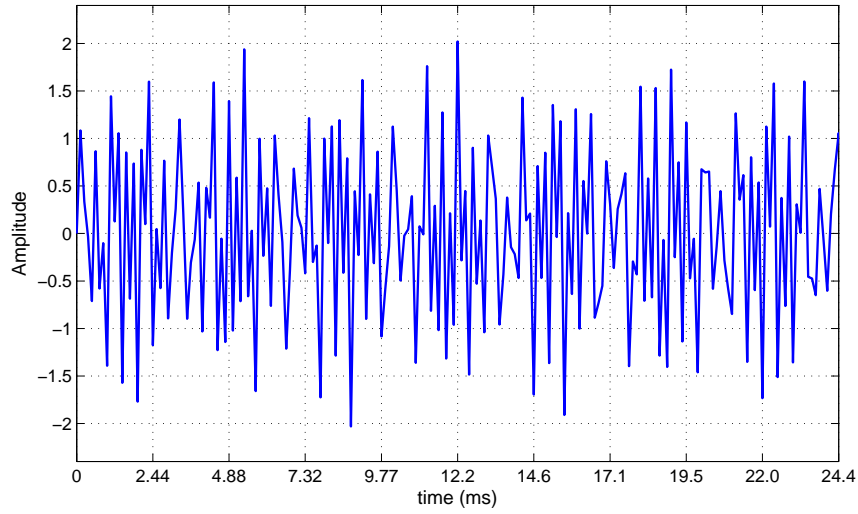


Figure 6.4. Recovered Signal.

6.2.2 Compressive Sensing for Frequency Domain Recovery

The result in section 6.2.1 shows the ability of compressive sensing to recover a sparse signal in the frequency domain, and then apply the inverse Fourier transform in order to recover the original time domain signal. The time domain signal is not always needed. A cognitive radio typically does not need to decode the signal, as discussed in section 3, it just needs to know which frequencies are being occupied. Thus, for the duration of this study, the focus is on the performance and limitations for recovering noisy signals in the frequency domain.

In order to simulate compressive sensing in varying SNR environments a program is written in Python that utilizes GNU Radio signal processing blocks. The blocks are used to generate a sparse signal in the frequency domain, and then collect and store m samples to a vector. The Python program then applies compressive sensing to the data stored in the vector.

The Python program applies compressive sensing following a similar setup to that used in 6.2.1. The measurement matrix, A , is again constructed by selecting the

corresponding n rows of the inverse FFT matrix. In the presence of noise, the compressive sensing problem is reformulated as given in equation 6.6. The l_1 -norm is still minimized, however the problem is now constrained by the l_2 -norm of $Ax - y$ being smaller than a constraint, ϵ . An adequate value of the constraint is given in [30] as

$$\epsilon^2 = \sigma^2(n + \lambda\sqrt{n}) , \quad (6.7)$$

where σ^2 is the noise standard deviation, and λ is a constant. For the results in this paper, $\lambda = 2$. In order to recover the signal, a Python CVX l_1 minimization solver in Python from [34] is used.

The setup of the experiment to study the compressive sensing at varying SNR is as follows. A sparse frequency domain signal is created by generating and adding two sinusoidal signals with frequencies of -10 kHz and 12 kHz, each with an amplitude of 1. Gaussian random noise is also added to the signals to simulate different SNR values. The SNR is varied by changing the Amplitude/standard deviation of the noise source. The SNR is calculated in this study using

$$SNR = 20 \log_{10} \frac{A_{Signal}}{A_{Noise}} , \quad (6.8)$$

where A_{Signal} is the amplitude of the sinusoids, and A_{Noise} is the amplitude or the standard deviation of the noise source. Hence $A_{Noise} = \sigma$. The sampling rate used is 32000 complex (IQ) samples per second. Based on Nyquist theory this results in a detectable bandwidth of 32 kHz, spanning from -16 kHz to 16 kHz. Figure 6.5 shows a block diagram in GRC used to generate the signal and collect the samples for the simulations.

The size, m , of the original time domain signal, and the size of the recovered frequency domain signal, x , is held constant at $m = 1024$ for this study. The number of time domain samples, n , used to recover the signal was varied at values between 64 and 868 samples with the noise source disabled. The program was able to perfectly recover the signal at all of the values. As the number of samples increased, the recovery time

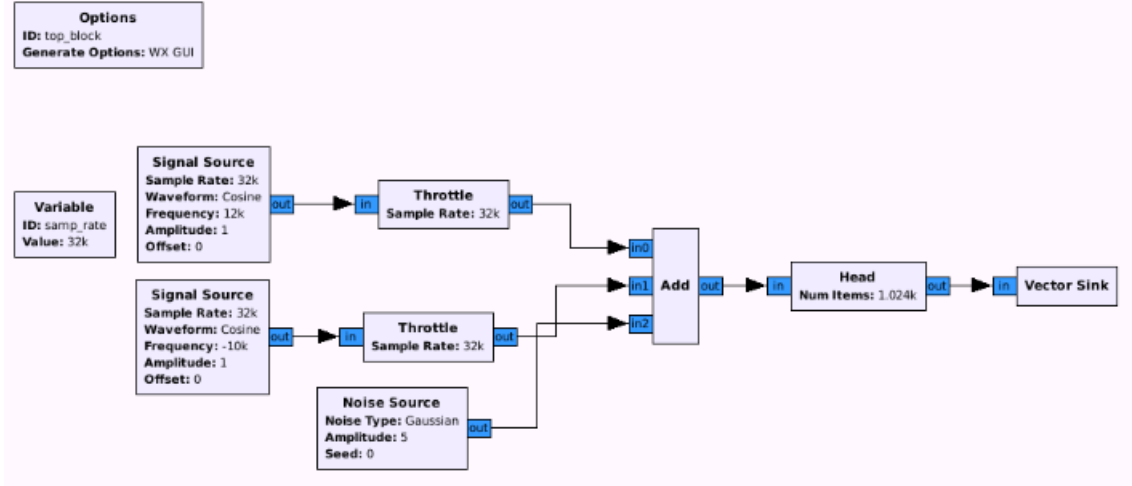


Figure 6.5. Simulation signal generation block diagram.

decreased. Thus, $n = 512$ was chosen as a good tradeoff between recovery time and compression ratio.

With the $m = 1024$ and $n = 512$, compressive sensing is then applied to the signal at different SNR values. The constraint for the l_1 solver given in equation 6.7 is adjusted accordingly for the noise variance in each scenario. The accuracy of the recovered frequency domain signal is measured by computing and plotting the 1024 point FFT and visually comparing it to the recovered frequency domain signal, and also analytically comparing it to the recovered spectrum. This is done analytically by comparing the percent error magnitudes of the FFT points at frequencies -10 and 12 kHz of the original and recovered signal. The percent error is given by

$$X_{f-error} = 100 \cdot \frac{|X_{f-recovered} - X_{f-original}|}{X_{f-original}}, \quad (6.9)$$

where $X_{f-recovered}$ is the magnitude of the recovered point in the spectrum and $X_{f-original}$ is the magnitude of the original spectrum point at 10 or 12 kHz. The dynamic range is also measured for 10 and 12 kHz to show its robustness to noise, which is given by

$$D.R.f = X_{f-recovered} - X_{Noise}, \quad (6.10)$$

where X_{Noise} is the magnitude of the largest noise component, and f corresponds to the frequency of -10 kHz or 12 kHz.

Table 6.1 shows the results when compressive sensing is applied to the signal at varying SNR with a 50% compression ratio. The recovery took approximately 90 seconds for all of the scenarios. It should be noted that the recovered spectrum is compared to the computed FFT of the spectrum in the presence of the respective amount of noise. Figure 6.6 and 6.7 show the original and recovered spectrum respectively when the SNR is 1.25. Figure 6.8 and 6.9 shows the original and recovered spectrum respectively when the SNR is -3.25 dB.

The largest resulting error for the range of SNR's covered is 70.39% when the SNR is -8.79 dB. This is relatively large, however the dynamic range is 278. Figure shows the a plot of the recovered spectrum when the SNR is -8.79 dB. Based on the dynamic range and as shown in the figure, the location of the spike can still easily be resolved.

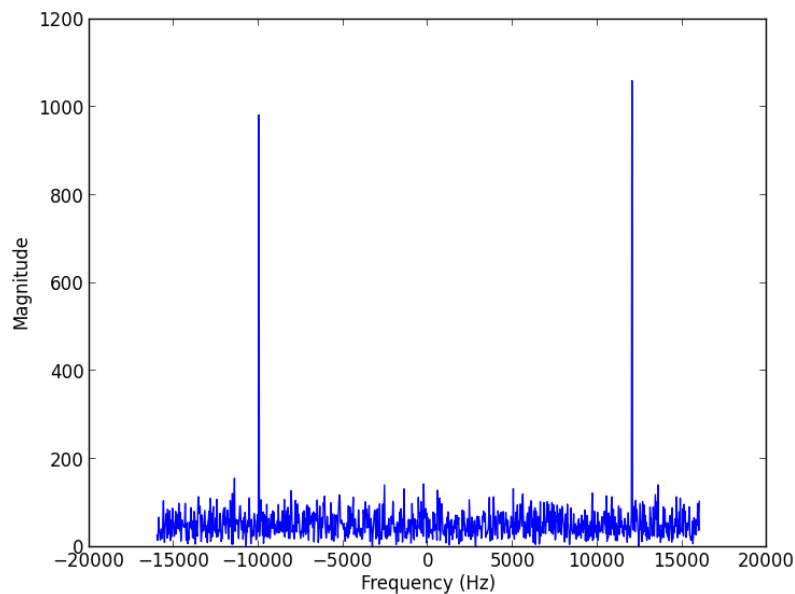


Figure 6.6. Original spectrum with SNR = -1.94 dB.

Compressive sensing was applied in the same scenario with SNR values below -10.23

Table 6.1. Compressive sensing performance with $m = 1024$ and $n = 512$.

A_{noise}, σ	SNR (dB)	$X_{(-10kHz)error}$ (%)	$X_{(12kHz)error}$ (%)	$D.R. - 10kHz$	$D.R. 12kHz$
0	∞	0	0	1024	1024
50m	26.02	1.27	1.17	1009	1014
200m	13.98	5.69	5.73	960	971
500m	6.02	7.44	5.11	918	970
750m	2.50	9.51	10.23	939	909
1.0	0	7.78	18.50	893	839
1.25	-1.94	22.91	22.03	700	779
1.75	-4.86	27.39	31.84	680	673
2.25	-7.04	32.28	20.82	585	776
2.75	-8.79	70.39	51.67	278	534
3.25	-10.23	62.12	43.42	300	582

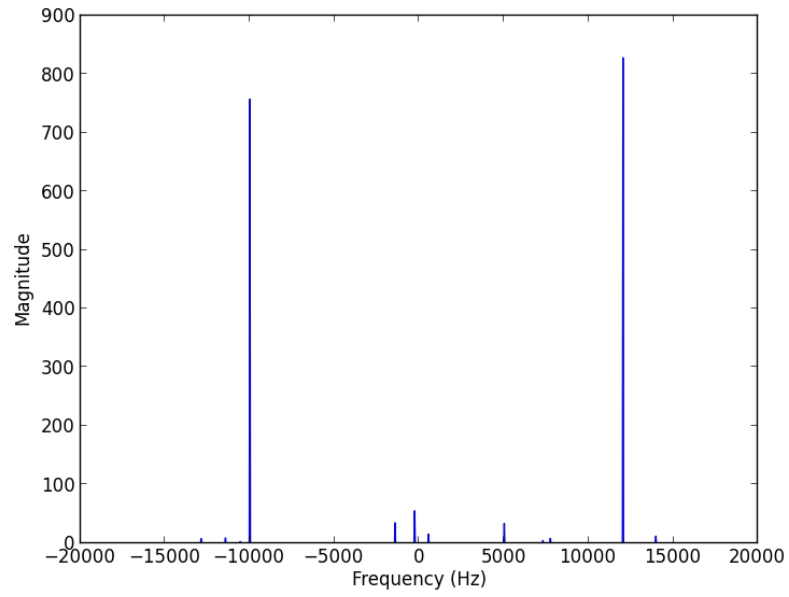


Figure 6.7. Recovered spectrum with $\text{SNR} = -1.94$ dB and $n = 512$.

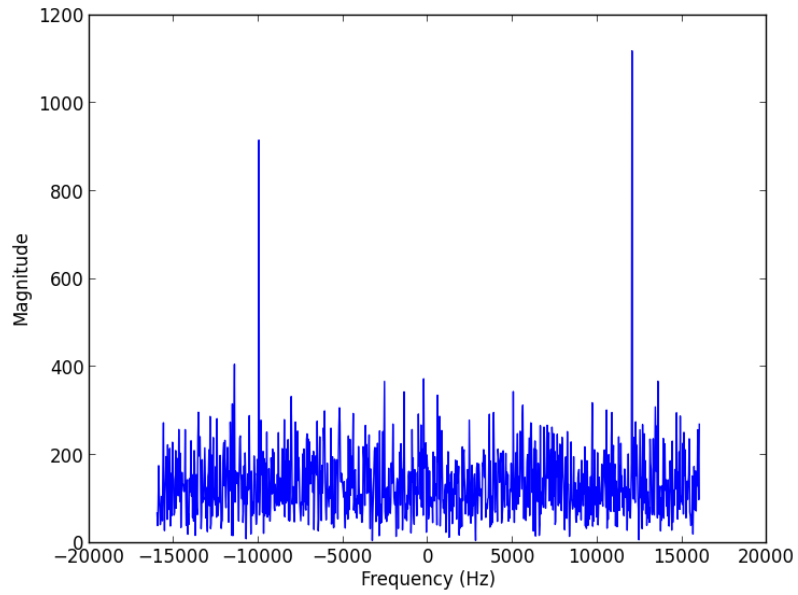


Figure 6.8. Original spectrum with $\text{SNR} = -10.23$ dB.

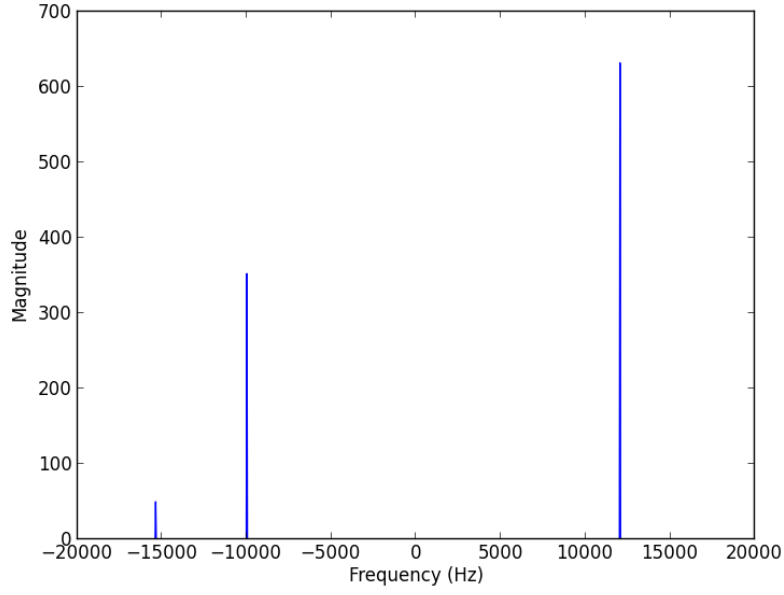


Figure 6.9. Recovered spectrum with SNR = -10.23 dB and $n = 512$.

dB, however the signal was not able to be consistently recovered. At low SNR, the number of samples used for signal recovery can be increased to increase the accuracy of the recovered signal. The number of samples n is increased to 768 and compressive sensing is again applied to recover the signal. Table 6.2 shows the results of the recovery at lower SNR values. Increasing the number of samples results in an increase in overall recovery time, the results in for $n = 768$ took approximately 140 seconds to recover the signal.

Figure 6.10 shows the recovered spectrum when the SNR is -3.25 dB, the accuracy is greater than that of figure 6.9 when only 512 samples are used. With n increased to 768, compressive sensing was able to accurately recover the spectrum down to -13.06 dB. Figures 6.11 and 6.12 shows the original and recovered spectrum respectively at an SNR of -13.06 dB. As the SNR falls, the overall accuracy decreases, however the location of the spikes can be still be recovered. One possible way to recover the location of the spike in practice, would be to apply energy detection to each FFT bin. With a dynamic range of 291, the energy detector would be able to detect the spike with high probability.

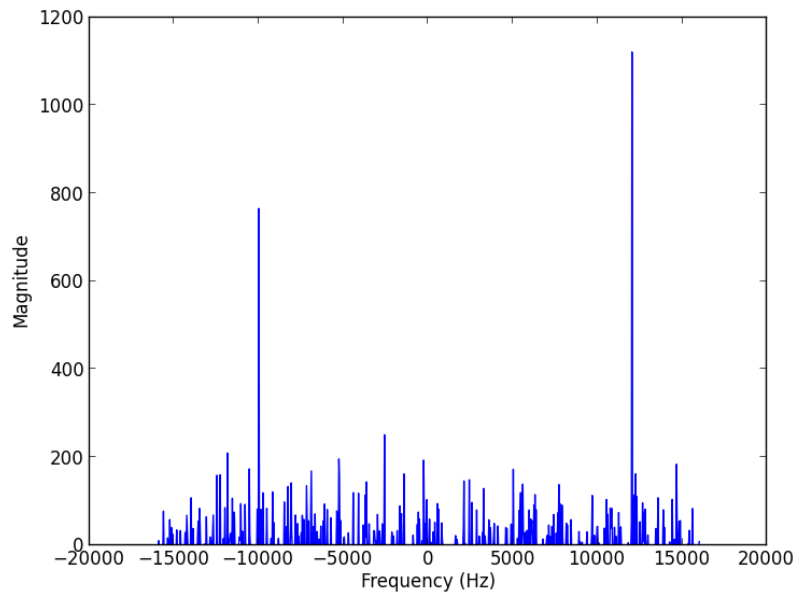


Figure 6.10. Recovered spectrum with $\text{SNR} = -10.23$ dB and n is increased to 768.

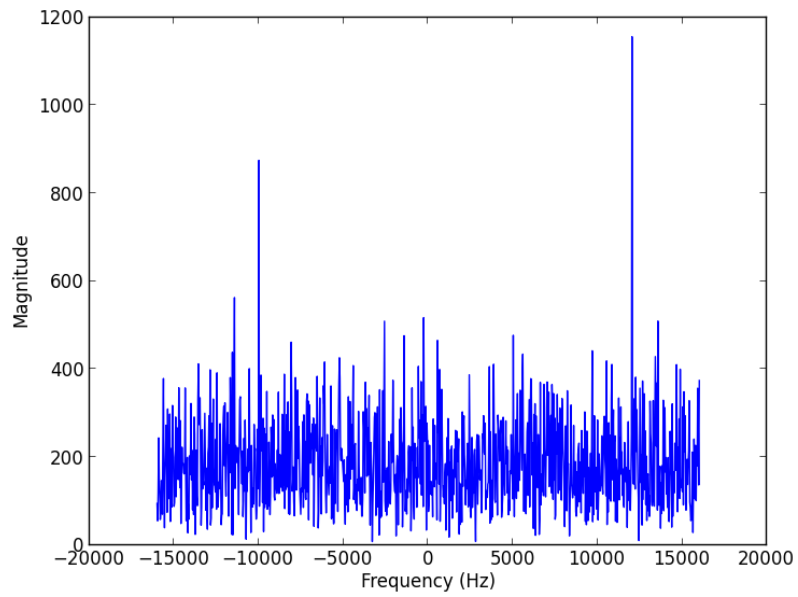


Figure 6.11. Original spectrum with $\text{SNR} = -13.06$ dB.

Table 6.2. Compressive sensing performance with $m = 1024$ and $n = 768$.

A_{noise}, σ	SNR (dB)	$X_{(-10kHz)error}$ (%)	$X_{(12kHz)error}$ (%)	$D.R. - 10kHz$	$D.R. 12kHz$
3.25	-10.23	22.14	3.29	510	864
3.6	-11.13	20.58	25.62	469	603
4.0	-12.04	12.96	17.83	491	652
4.5	-13.06	26.26	30.42	291	453

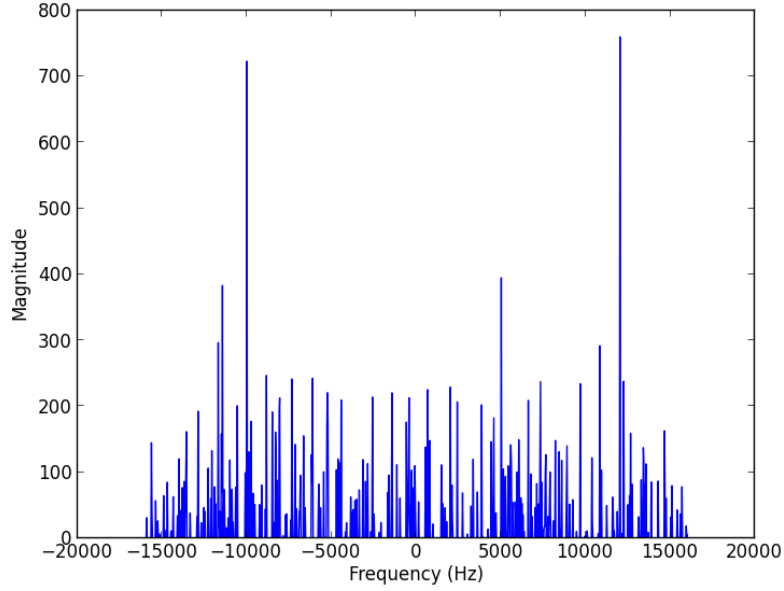


Figure 6.12. Recovered spectrum with $SNR = -13.06$ dB and $n = 768$.

6.3 COMPRESSIVE SENSING IMPLEMENTATION

Compressive sensing is then implemented using the USRP's. The experimental setup is similar to that of the simulations. Two sinusoidal signals are transmitted, compressive sensing is then applied at the receiving end.

A USRP B210 and an ultra wideband antenna that are connected to a laptop computer acts as the receiver for these experiments. More information on the antenna

can be found in [35]. The same Python program from the simulations is used to perform the compressive sensing. The only major difference is that rather than generating the signal using GNU Radio blocks, the USRP receives the signals wirelessly. This is implemented into the program by adding a GNU Radio block that interfaces with the computer with the USRP. The number collect, as well as the sampling and center frequencies for the USRP are also done using GNU Radio functions.

The experiment is done in an anechoic chamber in order to use a transmitter with known SNR. The experiment is done at relatively low transmission power, so the chamber also provides isolation from outside interference, and also isolates the USRP transmission from causing any harmful interference to other devices.

Since two sinusoidal signals are transmitted, two separate transmitters are used and they each transmit one sinusoid. A USRP B210 connected to a separate laptop acts as the first transmitter. A network analyzer signal generator and waveguide is used as the second transmitter.



Figure 6.13. Experimental setup.

The sampling frequency of the receiving USRP is set to 10.666667M complex (IQ) samples per second. Hence, the bandwidth of the recovered spectrum is also 10.666667 MHz. For this experiment, the center frequency of the USRP is tuned to 4.996667 GHz. The spectrum analyzer is tuned to transmit a continuous sinusoid at 5 GHz, and other transmitting USRP is set to transmit a continuous sinusoid at 4.994667 GHz. Thus, the transmitted signals are approximately 5.333 MHz apart. In the frequency domain the signals will appear as two continuous spikes a 4.994667 GHz and 5 GHz.

Following the same setup as the simulations, $m = 1024$ samples are collected by the receiving USRP and $n = 512$ samples are used to apply recover the signal. Figure 6.14 shows the FFT recovered spectrum of the collected samples from the USRP. A large spike is present at the center frequency of the plot. The center frequency corresponds to a zero baseband frequency, so the spike is due to a DC bias. Figure 6.15 shows the recovered spectrum when compressive sensing is applied. The constraint is adjusted in order to get more accurate results. 6.16 shows the recovered spectrum with a tighter constraint for the solver.

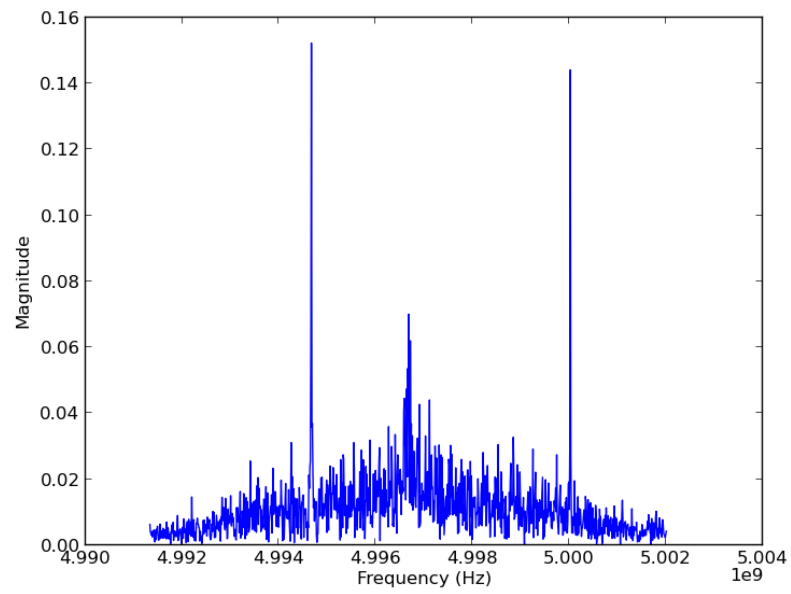


Figure 6.14. Original spectrum.

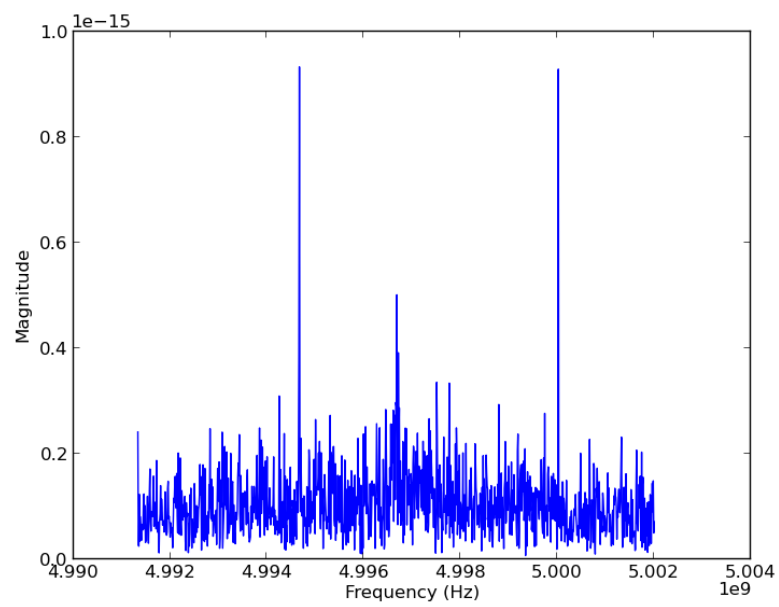


Figure 6.15. Recovered spectrum.

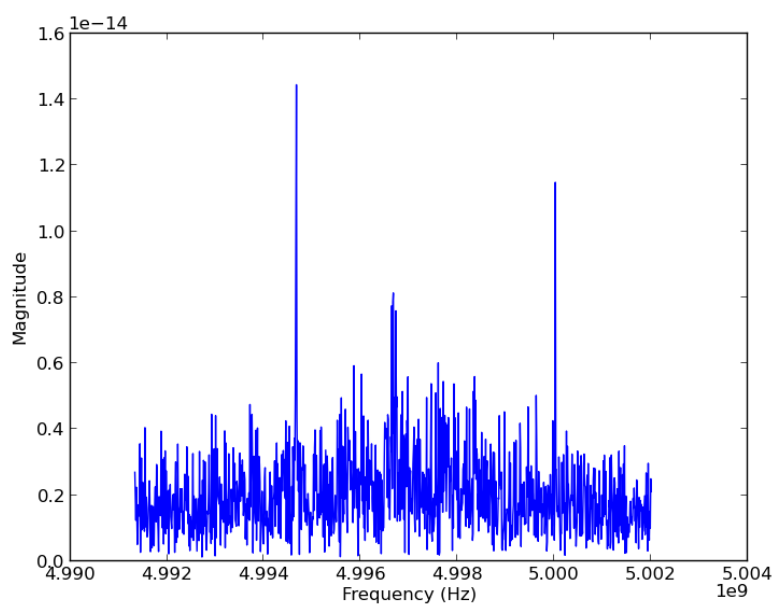


Figure 6.16. Recovered spectrum.

CHAPTER 7

CONCLUSION

Compressive sensing and energy detection are both viable detection methods when no knowledge of the signal of interest is known. As shown from the results, the performance of both is limited at low SNR. Compressive sensing uses far fewer samples than that required by Nyquist sampling theorem, giving it the potential to accurately recover a much wider bandwidth. Energy detection is simple and efficient with narrow bands, however, for wide band sensing it requires high rate ADC's or filter banks, which take space and can be expensive.

The effectiveness of Python and GNU Radio for simulation and implementation was demonstrated in this study. When a simulation is set up using Python and GNU Radio, it can then be implemented to the USRP by just switching out a few blocks of code. This can simplify designs and save time.

The USRP is an effective means of testing the performance of spectrum sensing schemes. Having been the first work done on the USRP's in the Telecommunications lab, this work can serve as a guide for future students to build upon in the area of spectrum sensing and or compressive sensing for cognitive radios. It can also serve as a guide to help future students implementing and testing designs for other research topics. The compressive sensing results can be expanded to study the recovery performance for different types of signals with different levels of sparsity. The constraint can also be varied to examine the tradeoff between recovery time and accuracy of the recovery. When SNR is relatively high, it is likely that lowering the constraint will reduce recovery time, while also lowering the recovery accuracy. The locations of the occupied frequencies should still be able to be easily determined up to a certain point. Another wide band detection method such as wavelet based detection or sparse fast Fourier transform can also be a focus of future implementation.

This study focused on using Python and GNU Radio for the USRP. Future work can utilize the FPGA to implement custom designs onto the USRP. Custom programs can also be implemented using C++ and GNU Radio rather than Python. This can allow better performance and also make it easier for designs to be reused by building custom C++ blocks that can then be reused in a Python script.

REFERENCES

- [1] M.A. Mchenry, D. McCloskey & D. Roberson, "Spectrum Occupancy Measurements Chicago, Illinois November 16-18, 2005," Dec. 20, 2005. [Online].
- [2] J. Mitola; G.Q. Maguire, Jr., "Cognitive radio: making software radios more personal," *Personal Communications, IEEE* , vol.6, no.4, pp.13,18, Aug 1999
- [3] Federal Communications Commission "Notice of Proposed Rule Making and Order," *ET Docket No.03-322m* Dec. 30, 2003
- [4] Y. Tachwali; B.F. Lo.; I.F. Akyildiz, ; R. Agusti, "Multiuser Resource Allocation Optimization Using Bandwidth-Power Product in Cognitive Radio Networks," *Selected Areas in Communications, IEEE Journal on*, vol.31, no.3, pp.451,463, March 2013
- [5] Jui-Chi Liang; Jyh-Cheng Chen, "Resource Allocation in Cognitive Radio Relay Networks," *Selected Areas in Communications, IEEE Journal on* , vol.31, no.3, pp.476,488, March 2013
- [6] L. Gao; P. Wu; S. Cui, "Power and Rate Control with Dynamic Programming for Cognitive Radios," *Global Telecommunications Conference, 2007. GLOBECOM '07.* IEEE, pp.1699,1703, 26-30 Nov. 2007
- [7] Z. Han & K.J. Liu, "Resource allocation for wireless networks. Basics, techniques, and applications", *New York: Cambridge University Press*, 2008
- [8] A. Feickert "The Joint Tactical Radio System (JTRS) and the Armys Future Combat System (FCS): Issues for Congress," *CRS Report for Congress*, November 2005
- [9] L. Lu, X. Zhou, U. Onunkwo, G.Y. Li "Ten years of research in spectrum sensing and sharing in cognitive radio," *EURASIP Journal on Wireless Communications and Networking*, 2012, 2012:28
- [10] C. Stevenson, G. Chouinard, L. Z. Lei, W. Hu, S.J. Shellhammer, W. Caldwell, "IEEE 802.22: The first cognitive radio wireless regional area network standard,"

Communications Magazine, IEEE, vol. 47, no.1, pp.130,138,January 2009

- [11] C.S. Sum; G.P. Villardi; M.A. Rahman; T. Baykas; H.N. Tran; Z. Lan; C. Sun; Y. Alemseged; J. Wang; C. Song; C.W. Pyo; S. Filin; H. Harada “Cognitive communication in TV white spaces: An overview of regulations, standards, and technology [Accepted From Open Call],” *Communications Magazine, IEEE* , vol.51, no.7, pp.138,145, July 2013
- [12] T. Yucek; H. Arslan, “A survey of spectrum sensing algorithms for cognitive radio applications,” *Communications Surveys & Tutorials, IEEE* , vol.11, no.1, pp.116,130, First Quarter 2009
- [13] W.H. Chung, “Sequential Likelihood Ratio Test under Incomplete Signal Model for Spectrum Sensing,” *Wireless Communications, IEEE Transactions on* , vol.12, no.2, pp.494,503, February 2013
- [14] Y. Xin; H. Zhang; L. Lai, “A Low-Complexity Sequential Spectrum Sensing Algorithm for Cognitive Radio,” *Selected Areas in Communications, IEEE Journal on* , vol.32, no.3, pp.387,399, March 2014
- [15] S.J. Kim; G.B. Giannakis, “Sequential and Cooperative Sensing for Multi-Channel Cognitive Radios,” *Signal Processing, IEEE Transactions on* , vol.58, no.8, pp.4239,4253, Aug. 2010
- [16] A. Bagwari, G.S. Tomar, S. Verma, “Cooperative Spectrum Sensing Based on Two-Stage Detectors With Multiple Energy Detectors and Adaptive Double Threshold in Cognitive Radio Networks,” *Electrical and Computer Engineering, Canadian Journal of* , vol.36, no.4, pp.172,180, Fall 2013
- [17] K. Letaief, W. Zhang, “Cooperative Communications for Cognitive Radio Networks,” *Proceedings of the IEEE* , vol.97, no.5, pp.878,893, May 2009
- [18] Y.E. Lin, K.H. Liu, H.Y. Hsieh, “On Using Interference-Aware Spectrum Sensing for Dynamic Spectrum Access in Cognitive Radio Networks,” *Mobile Computing, IEEE Transactions on* , vol.12, no.3, pp.461,474, March 2013

- [19] D. Cabric; S.M. Mishra; R.W. Broderen, "Implementation issues in spectrum sensing for cognitive radios" *Signals, Syst. and Comput. Conference*, 2004
- [20] Z. Ling-ling, H. Jian-guo, T. Cheng-kai, "Novel energy detection scheme in cognitive radio," *Signal Processing, Communications and Computing (ICSPCC), 2011 IEEE International Conference on* , vol., no., pp.1,4, 14-16 Sept. 2011
- [21] Y. Chen, "Improved energy detector for random signals in gaussian noise," *Wireless Communications, IEEE Transactions on* , vol.9, no.2, pp.558,563, February 2010
- [22] H. Sun; A. Nallanathan; C.X. Wang; Y. Chen, "Wideband spectrum sensing for cognitive radio networks: a survey," *Wireless Communications, IEEE*, vol.20, no.2, pp., April 2013 .12, pp.5406,5425, Dec. 2006
- [23] B. Farhang-Boroujeny, "Filter Bank Spectrum Sensing for Cognitive Radios," *Signal Processing, IEEE Transactions on* , vol.56, no.5, pp.1801,1811, May 2008
- [24] T. Sato; M. Umehira, "A new spectrum sensing scheme using overlap FFT filter-bank for dynamic spectrum access," *Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM), 2011 Sixth International ICST Conference on* pp.6,10, 1-3 June 2011
- [25] Huseyin, A., ed. *Cognitive radio, software defined radio, and adaptive wiereless systems*. Vol. 10. Berlin: Springer, 2007.
- [26] <http://gnuradio.org/redmine/projects/gnuradio/wiki>
- [27] M.A. Sarijari; A. Marwanto; N. Fisal; S.K. Yusof; R.A. Rashid; M.H. Satria, "Energy detection sensing based on GNU radio and USRP: An analysis study," *Communications (MICC), 2009 IEEE 9th Malaysia International Conference on* pp.338,342, 15-17 Dec. 2009
- [28] R.A. Rashid, M. Sarijari, N. Fisal, S.K.S. Yusof, N.H. Mahalin, "Spectrum Sensing Measurement using GNU Radio and USRP Software Radio Platform", *Wireless and Mobile Communications, 2011 IEEE 7th Internaitonal Conference on*, pp.237,242, 23-25 Sep.

- [29] Y.C. Eldar & G. Kutyniok, "Introduction to compressed sensing," from *Compressed Sensing*, Cambridge, UK; Cambridge University Press, pp. 1-64.
- [30] E.J. Candes, J.K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics*, vol. 59, pp. 1207-1223, 2006.
- [31] E.J. Candes, J.K. Romberg, and T. Tao, "Decoding by linear programming," *IEEE Trans. on Inform. Theory*, vol. 51, pp. 4203-4215, 2005.
- [32] E.J. Candes, T. Tao, "Near-optimal signal recovery from random projections: Univesal Encoding Strategies," *Information Theory, IEEE Transactions on*, vol.52, no
- [33] E. Candes and J. Romberg, "L1-magic: recovery of sparse signals via convex programming", October 2005
- [34] M. Mahmudimanesh, "General purpose norm-1 minimization solver in Python", sourceforge.net/projects/pyl1min
- [35] H.R. Gorla, K. Shaw, D. Addison, F. Harackiewicz, "Ultra wideband dual square trident planar antenna," *Antennas and Propagation and USNC-URSI Radio Science Meeting, 2014 IEEE International Symposium on*, [Accepted] July 6-11.
- [36] B. Wang; K.J.R. Liu, "Advances in cognitive radio networks: A survey," *Selected Topics in Signal Processing, IEEE Journal of* , vol.5, no.1, pp.5,23, Feb. 2011
- [37] I.F. Akyildiz; W.Y. Lee; M.C. Vuran; S. Mohanty, "NeXt generation/dynamic spectrum access/cognitive Radio Wireless Networks: A Survey," *Computer Networks Journal*, vol. 50, pp. 2127-2159, 2006.

VITA

Graduate School
Southern Illinois University

Jason Stegman

jstegs94@gmail.com

Southern Illinois University at Carbondale
Bachelor of Science, Electrical Engineering, December 2012

Thesis Title:

Wideband and Narrowband Spectrum Sensing Methods Using Software Defined Radios

Major Professor: Dr. X. Zhou